EXPERIMENT NO: 1

Circular Convolution

Aim : To perform Circular Convolution on a given sequence using MATLAB software.

Software Used : MATLAB R2021b.

Program :

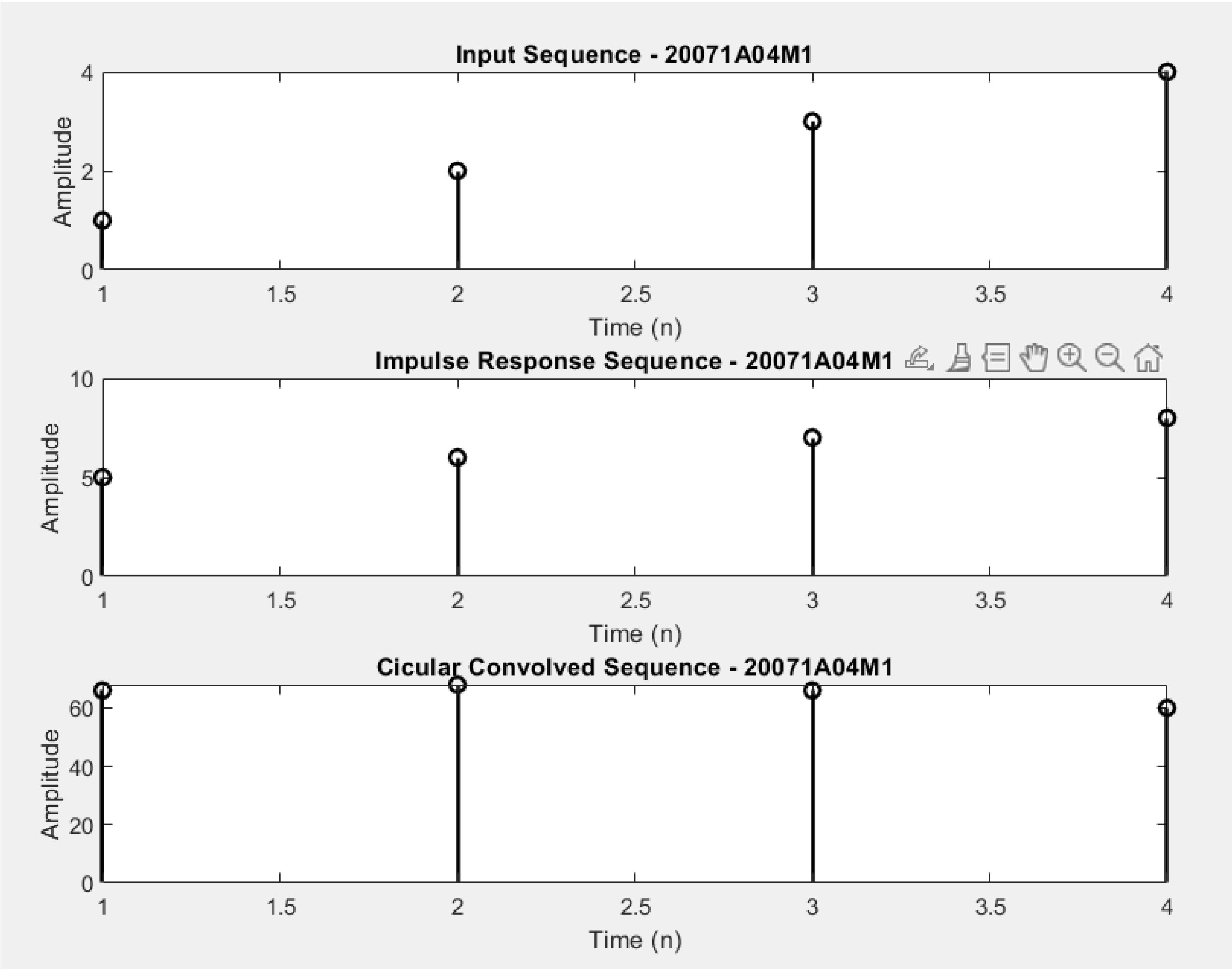
```
clc
clear all
x=[1 2 3 4];
h=[5 6 7 8];
m=length(x);
n=length(h);
N=max(m,n);
x=[x,zeros(1,N-m)];
h=[h,zeros(1,N-n)];
for n=1:N
y(n)=0;
for i=1:N
j=n-i+1;
if(j<=0)
j=N+j;
end
y(n)=(y(n)+x(i)*h(j));
end
end
disp(y)
%verification
x1=fft(x,n);
h2=fft(h,n);
p=x1.*h2;
```

```
c1=ifft(p);
disp(c1)
subplot(3,1,1)
stem(x,'k',LineWidth=1.5)
title("Input Sequence - 20071A04M1")
xlabel("Time (n)")
ylabel("Amplitude")
subplot(3,1,2)
stem(h,'k',LineWidth=1.5)
title("Impulse Response Sequence - 20071A04M1")
xlabel("Time (n)")
ylabel("Amplitude")
subplot(3,1,3)
stem(y,'k',LineWidth=1.5)
title("Cicular Convolved Sequence - 20071A04M1")
xlabel("Time (n)")
ylabel("Amplitude")
```

Output :

Y= 66 68 66 60

C1= 66 68 66 60



Result : Thus, Circular Convolution is performed on a given sequence.

EXPERIMENT NO: 2

Discrete Fourier Transform/ Inverse Discrete Fourier Transform

Aim : To find Discrete Fourier Transform and Inverse Discrete Fourier Transform on a given sequence using MATLAB software.

Software Used : MATLAB R2021b.

Program :

```
clc
clear all
x=[1 2 3 4 5];
N=length(x);
xk=zeros(1,N);
for i=1:N
w=2*pi*(i-1)/N;
for n=1:N
xk(i)=xk(i)+(x(n).*exp(-j*w*(n-1)));
end
end
disp(xk)
y=zeros(1,N);
for n=1:N
w=2*pi*(n-1)/N;
for i=1:N
y(n)=y(n)+(xk(i)*exp(j*w*(i-1)));
end
y(n)=abs(y(n))/N;
end
disp(y)
subplot(3,1,1)
stem(1:N,x,'k',LineWidth=1.5)
title("Input Sequence - 20071A04M1")
```

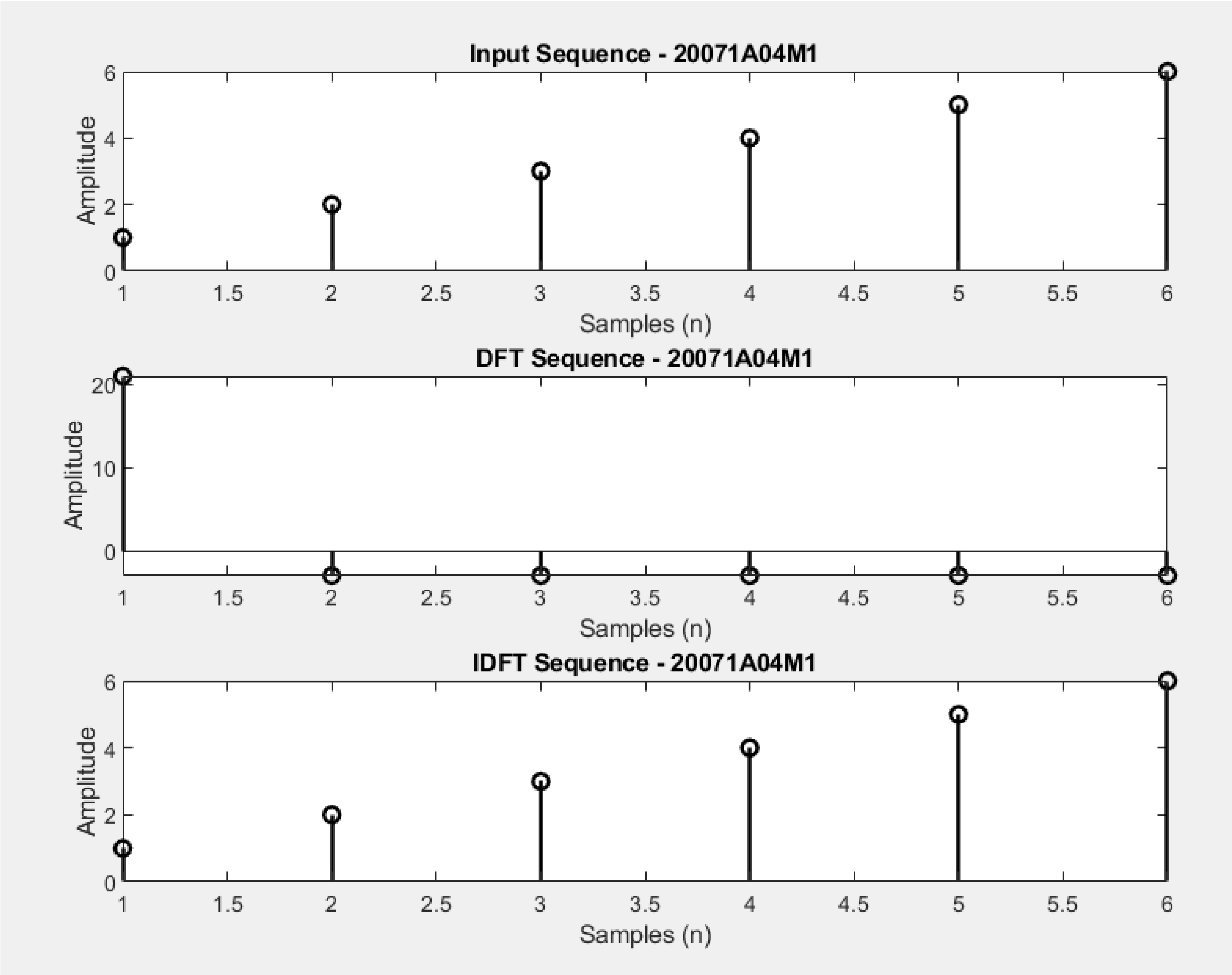
```
xlabel("Samples (n)")
ylabel("Amplitude")
subplot(3,1,2)
stem(1:length(xk),xk,'k',LineWidth=1.5)
title("DFT Sequence - 20071A04M1")
xlabel("Samples (n)")
ylabel("Amplitude")
subplot(3,1,3)
stem(1:length(y),y,'k',LineWidth=1.5)
title("IDFT Sequence - 20071A04M1")
xlabel("Samples (n)")
ylabel("Amplitude")
```

Output :

Xk= 21.0000 + 0.0000i , -3.0000 + 5.1962i, -3.0000 + 1.7321i,

-3.0000 - 0.0000i , -3.0000 - 1.7321i, -3.0000 - 5.1962i

y= 1.0000 2.0000 3.0000 4.0000 5.0000 6.0000



Result : Thus, Discrete Fourier Transform and Inverse Discrete Fourier Transform on a given sequence was found.

EXPERIMENT NO: 3

Power Spectrum Density

Aim : To find Power Spectrum Density on a given sequence using MATLAB software.

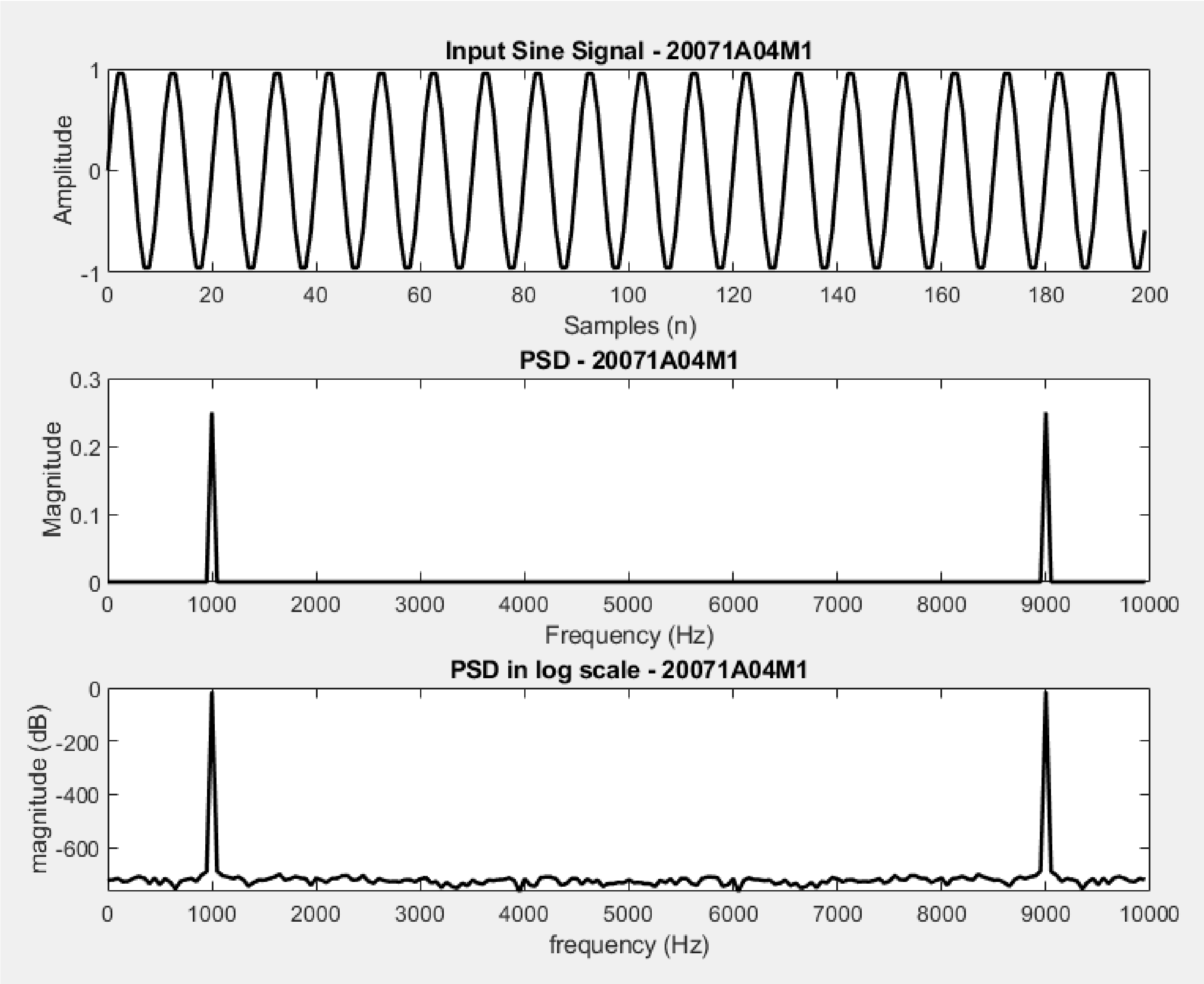
Software Used : MATLAB R2021b.

Program :

```
clc
clear all
fs=10000;
f1=1000;
N=200;
n=0:N-1;
x=sin(2*pi*f1*n/fs);
xk=abs(fft(x))/N;
p=xk.*xk;
f=(0:N-1)*fs/N;
subplot(3,1,1);
plot(n,x,'k',LineWidth=1.5);
title('Input Sine Signal - 20071A04M1');
xlabel('Samples (n)');
ylabel('Amplitude');
subplot(3,1,2);
plot(f,p,'k',LineWidth=1.5);
title('PSD - 20071A04M1');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
subplot(3,1,3);
plot(f,10*log(p),'k',LineWidth=1.5);
title('PSD in log scale - 20071A04M1');
xlabel('frequency (Hz)');
```

```
ylabel('magnitude (dB)');
```

Output :



Result : Thus, Power Spectrum Density on a given sequence was found.

EXPERIMENT NO: 4

Implementation of Filters using IIR

Aim : To implement High Pass and Low Pass Filters using IIR using MATLAB software.

Software Used : MATLAB R2021b.

Program :

a) High Pass Filter :

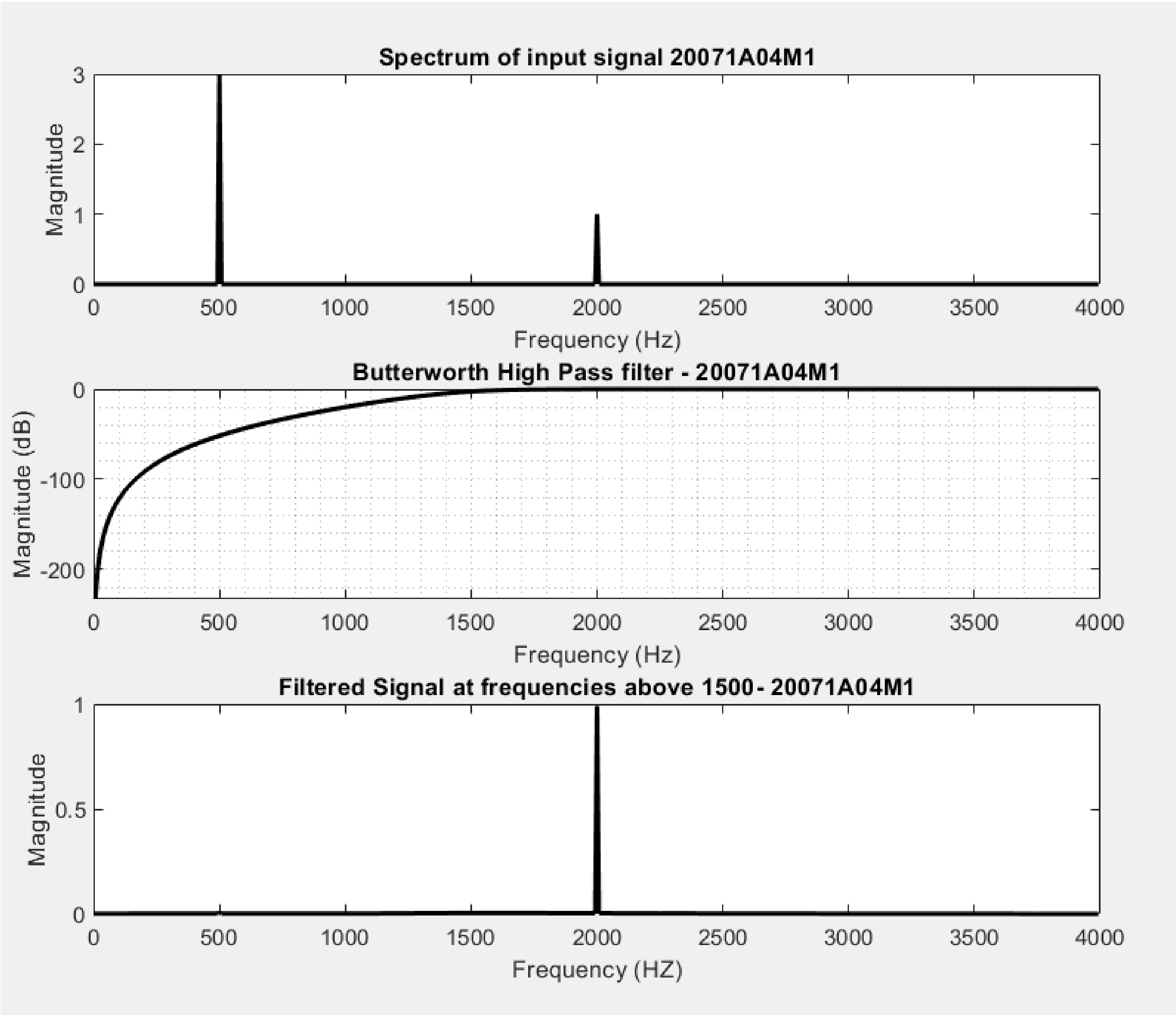
```
clc
clear all
fs=8000;
N=1024;
M=2;
n=0:N-1;
f1=500;
f2=2000;
x=3*sin(2*pi*f1*n/fs)+cos(2*pi*f2*n/fs);
X=2*abs(fft(x,N))/N;
X(1)=X(1)/2;
f=(0:1:N/2-1)*fs/N;
subplot(3,1,1)
plot(f,X(1:N/2),'k',LineWidth=1.5)
title('Spectrum of input signal 20071A04M1')
xlabel('Frequency (Hz)')
ylabel('Magnitude')
wp=1500;
ws=1000;
rp=3;
rs=20;
fF=fs/2;
```

```
wpa=wp/fF;
wsa=ws/fF;
[n,wc]=buttord(wpa,wsa,rp,rs);
[b,a]=butter(n,wc,'high');
[H,f_hz]=freqz(b,a,512,fs);
subplot(3,1,2)

plot(f_hz,20*log10(abs(H)),'k',LineWidth=1.5);
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
title('Butterworth High Pass filter - 20071A04M1');
grid minor
y=filter(b,a,x);
y=2*abs(fft(y,N))/N;
y(1)=y(1)/2;
f=(0:N/2-1)*fs/N;

subplot(3,1,3)
plot(f,y(1:N/2),'k',LineWidth=1.5)
title(['Filtered Signal at frequencies above ',num2str(wp),'-20071A04M1'])
xlabel('Frequency (HZ)')
ylabel('Magnitude')
```

Output :



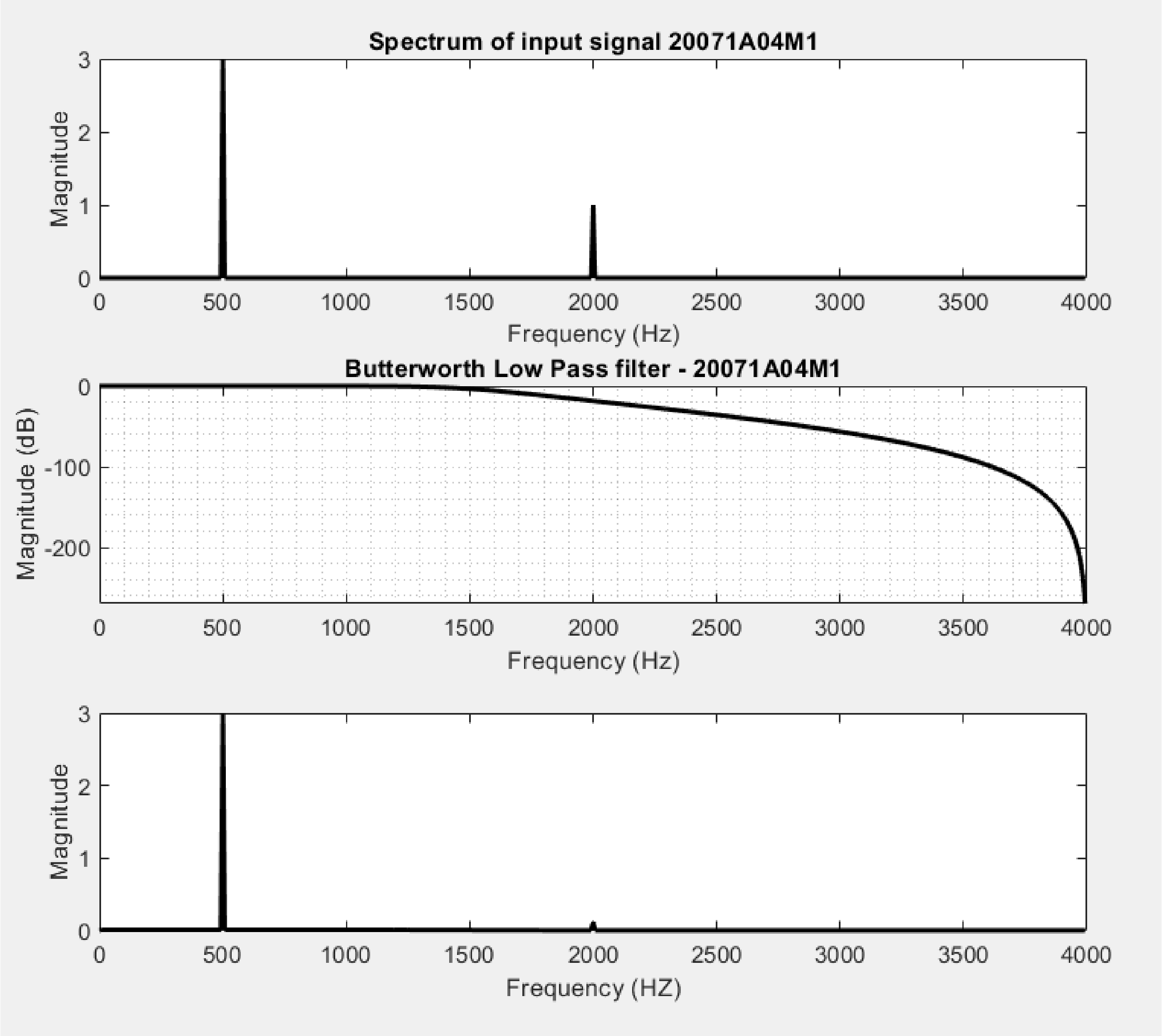
b) Low Pass Filter :

```
clc
clear all
fs=8000;
N=1024;
M=2;
n=0:N-1;
f1=500;
f2=2000;
x=3*sin(2*pi*f1*n/fs)+cos(2*pi*f2*n/fs);
```

```
X=2*abs(fft(x,N))/N;
X(1)=X(1)/2;
f=(0:1:N/2-1)*fs/N;
subplot(3,1,1)
plot(f,X(1:N/2),'k',LineWidth=1.5)
title('Spectrum of input signal 20071A04M1')
xlabel('Frequency (Hz)')
ylabel('Magnitude')
wp=1500;
ws=1000;
rp=3;
rs=20;
fF=fs/2;
wpa=wp/fF;
wsa=ws/fF;
[n,wc]=buttord(wpa,wsa,rp,rs);
[b,a]=butter(n,wc);
[H,f_hz]=freqz(b,a,512,fs);
subplot(3,1,2)
plot(f_hz,20*log10(abs(H)),'k',LineWidth=1.5);
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
title('Butterworth Low Pass filter - 20071A04M1');
grid minor
y=filter(b,a,x);
y=2*abs(fft(y,N))/N;
y(1)=y(1)/2;
f=(0:N/2-1)*fs/N;
subplot(3,1,3)
plot(f,y(1:N/2),'k',LineWidth=1.5)
xlabel('Frequency (HZ)')
```

```
ylabel('Magnitude')
```

Output :



Result : Thus, High Pass and Low Pass Filters using IIR are implemented.

EXPERIMENT NO: 5

Implementation of Filters using FIR

Aim : To implement High Pass and Low Pass Filters using FIR using MATLAB software.

Software Used : MATLAB R2021b.

Program :

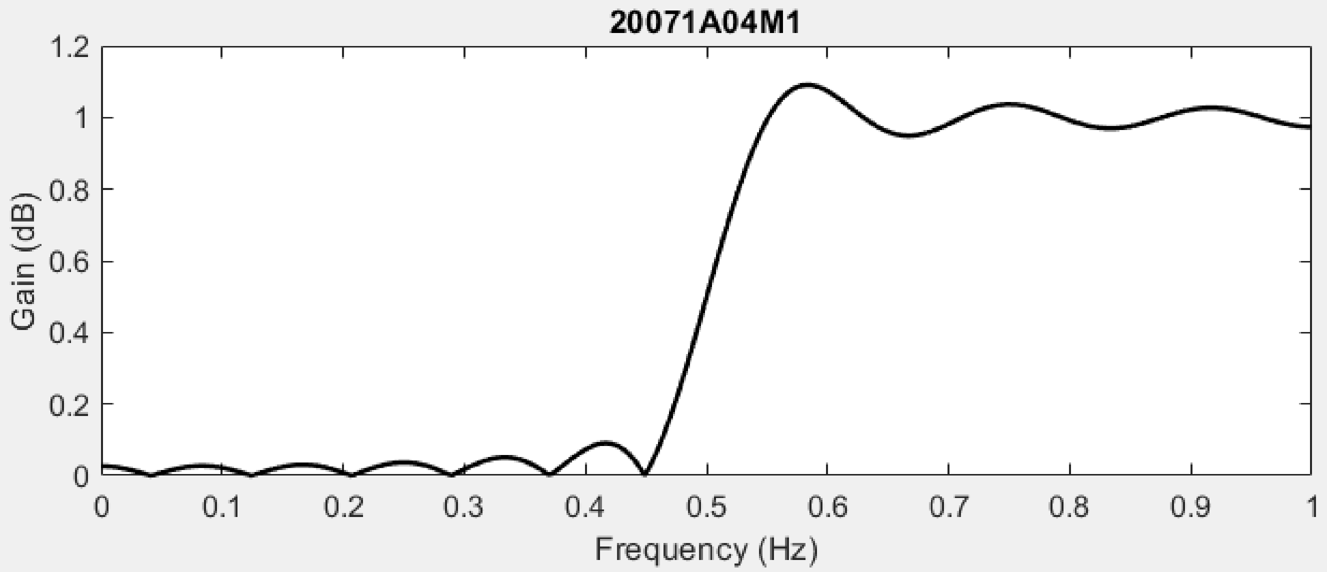
a) High Pass Filter :

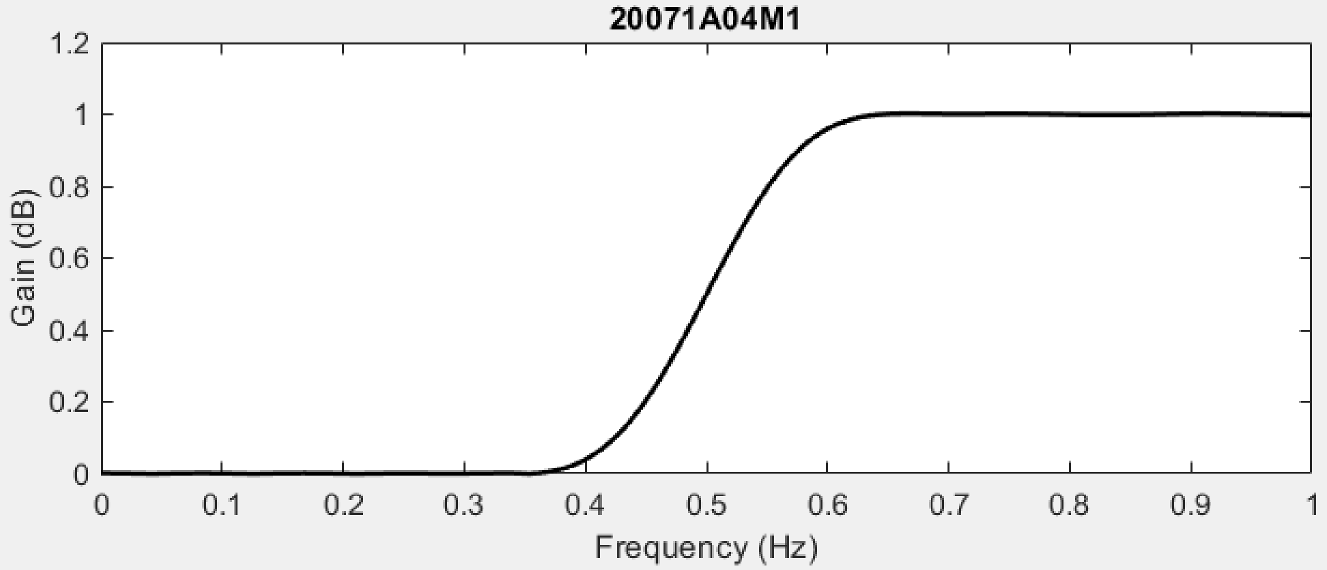
```
clc
clear all
wc=0.5*pi;
N=25;
alpha=(N-1)/2;
eps=0.001;
n=0:1:N-1;
hd=(sin(pi*(n-alpha+eps))-sin(wc*(n-alpha+eps)))./(pi*(n-alpha+eps));
wr=boxcar(N);
hn=hd.*wr';
w=0:0.01:pi;
h=freqz(hn,1,w);
subplot(2,1,1)
xlabel('Frequency (Hz)');
ylabel('Gain (dB)');
title('20071A04M1');

plot(w/pi,abs(h),'k',LineWidth=1.5)
wh=hamming(N);
hn=hd.*wh';
w=0:0.01:pi;
h=freqz(hn,1,w);
subplot(2,1,2)
plot(w/pi,abs(h),'k',LineWidth=1.5)


xlabel('Frequency (Hz)');
ylabel('Gain (dB)');
title('20071A04M1');
```

Output :



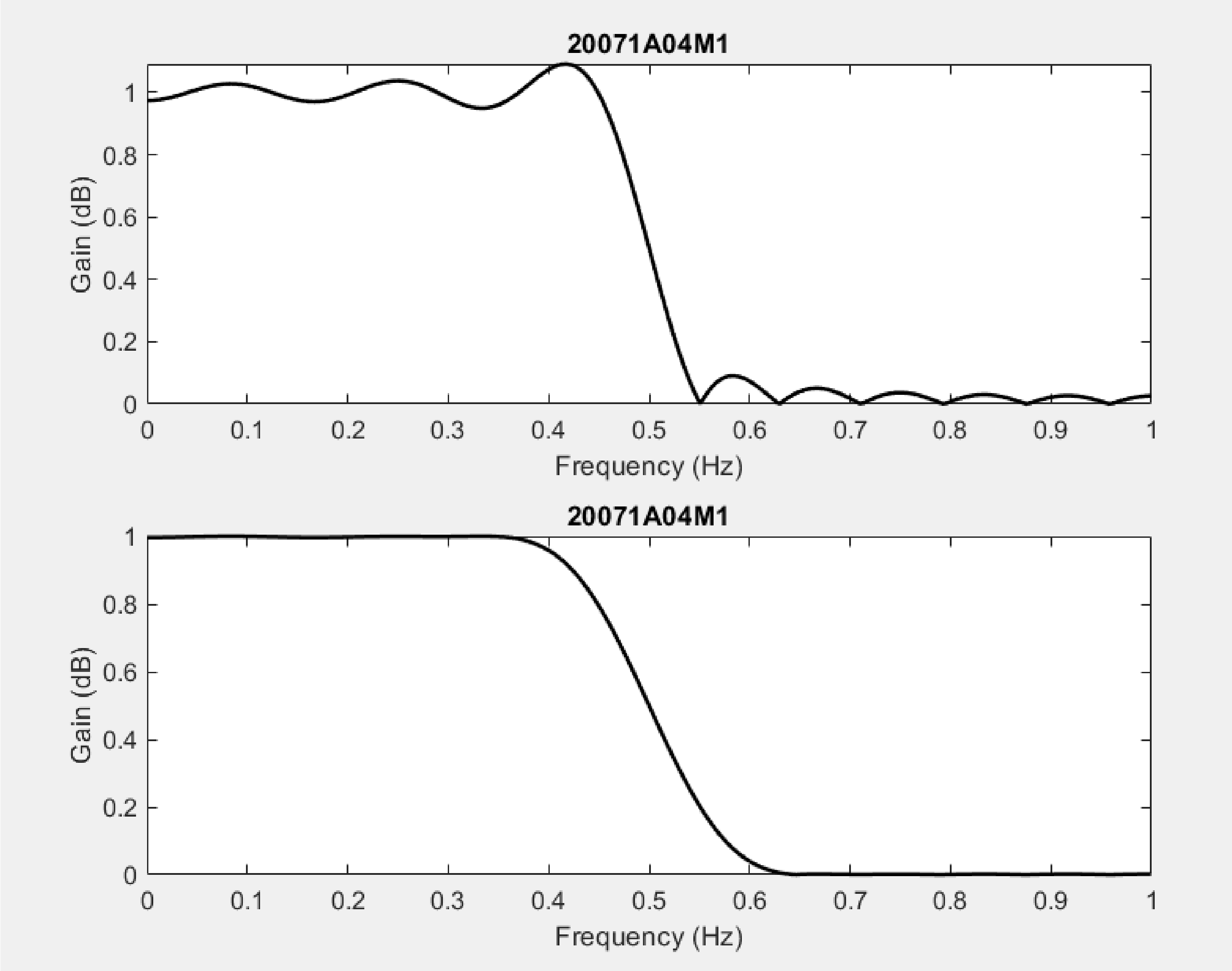


b) Low Pass Filter :

```
clc
clear all
wc=0.5*pi;
N=25;
```

```
alpha=(N-1)/2;
eps=0.001;
n=0:1:N-1;
hd=sin(wc*(n-alpha+eps))./(pi*(n-alpha+eps));
wr=boxcar(N);
hn=hd.*wr';
w=0:0.01:pi;
h=freqz(hn,1,w);
subplot(2,1,1)
plot(w/pi,abs(h),'k',LineWidth=1.5)
wh=hamming(N);
hn=hd.*wh';
w=0:0.01:pi;
h=freqz(hn,1,w);
subplot(2,1,2)
plot(w/pi,abs(h),'k',LineWidth=1.5)
```

Output :



Result : Thus, High Pass and Low Pass Filters using FIR are implemented.

EXPERIMENT NO: 6

Generation of Sinusoidal signal through filtering

Aim : To generate Sinusoidal Signal through filtering using MATLAB software.

Software Used : MATLAB R2021b.

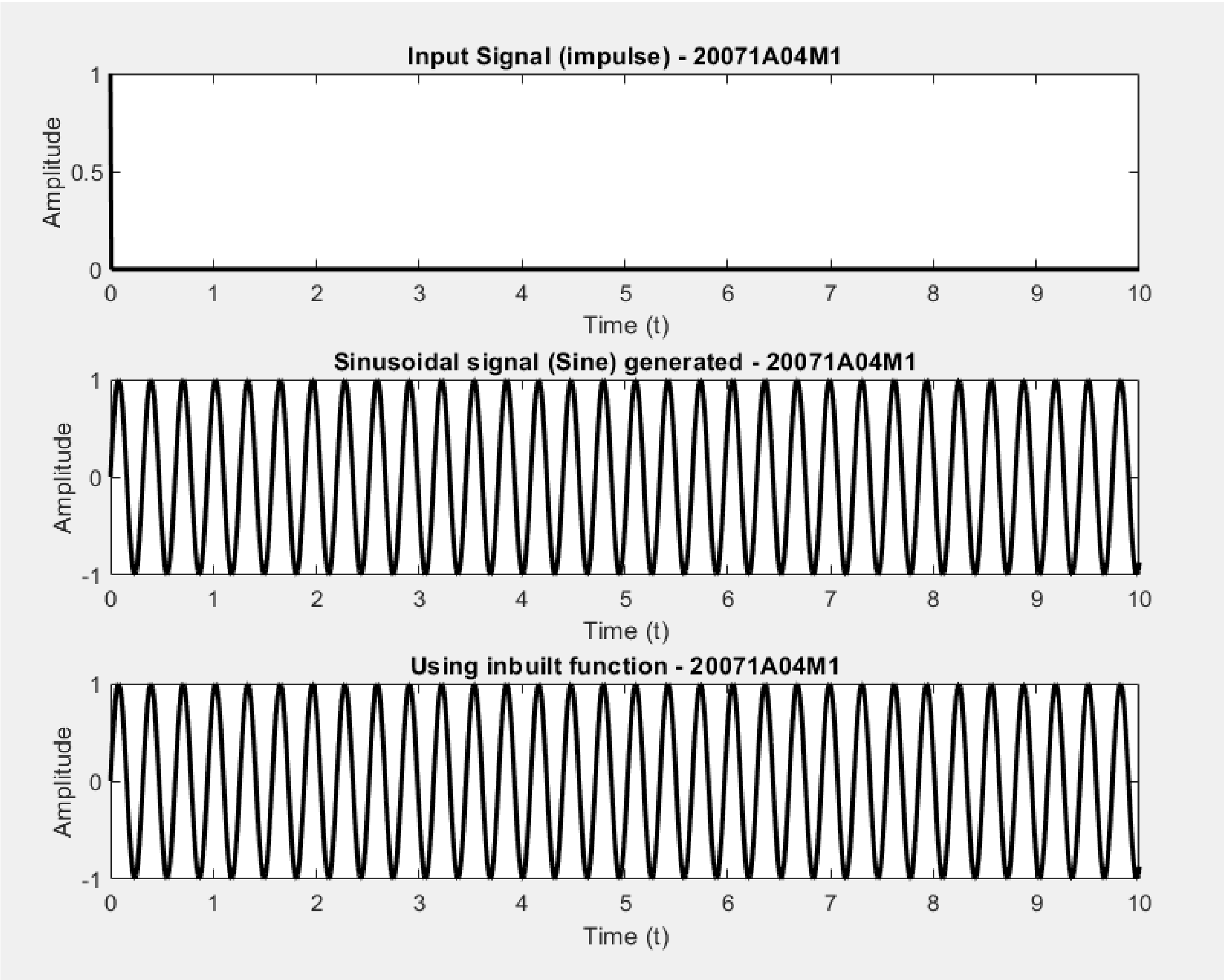
Program :

```
clc
clear all
t=0:0.01:10;
x=zeros(1,length(t));
x(find(t==0))=1;
w=input('Enter the value of w = ');
a=[1 -2*cos(w) 1];
b=[0 sin(w) 0];
y=zeros(1,length(t));
x1=0;
x2=0;
y1=0;
y2=0;
for n=1:length(x)
y(n)=b(1)*x(n)+b(2)*x1+b(3)*x2-a(2)*y1-a(3)*y2;
x2=x1;
x1=x(n);
y2=y1;
y1=y(n);
end
v=filter(b,a,x);
subplot(3,1,1)
plot(t,x,'k',LineWidth=1.5)
title('Input Signal (impulse) - 20071A04M1')
xlabel('Time (t)')
```

```
ylabel('Amplitude')
subplot(3,1,2)
plot(t,y,'k',LineWidth=1.5)
title('Sinusoidal signal (Sine) generated - 20071A04M1')
xlabel('Time (t)')
ylabel('Amplitude')
subplot(3,1,3)
plot(t,v,'k',LineWidth=1.5)
title('Using inbuilt function - 20071A04M1')
xlabel('Time (t)')
ylabel('Amplitude')
```

Output :

Enter the value of w = 0.2



Result : Thus, Sinusoidal Signal through filtering is generated.

EXPERIMENT NO: 7

Generation of DTMF Signals

Aim : To generate Dual Tone Multi Frequency Signal using MATLAB software.

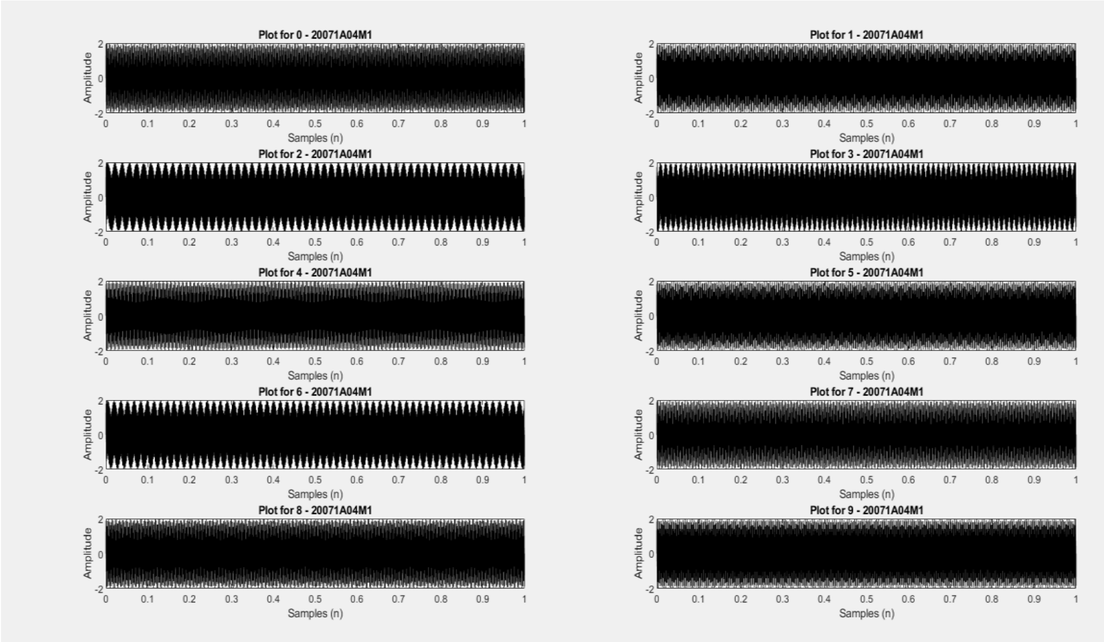
Software Used : MATLAB R2021b.

Program :

```
clc
clear all
num=input("Enter Number : ",'s')
fs=10000;
t=0:(1/fs):1;
x=2*pi*[697,770,852,941];
y=2*pi*[1209,1336,1477];
n=length(num)/2;
for i=1:length(num)
switch num(i)
case '1'
z=sin(x(1)*t)+sin(y(1)*t);
case '2'
z=sin(x(1)*t)+sin(y(2)*t);
case '3'
z=sin(x(1)*t)+sin(y(3)*t);
case '4'
z=sin(x(2)*t)+sin(y(1)*t);
case '5'
z=sin(x(2)*t)+sin(y(2)*t);
case '6'
z=sin(x(2)*t)+sin(y(3)*t);
case '7'
z=sin(x(3)*t)+sin(y(1)*t);
case '8'
```

```
z=sin(x(3)*t)+sin(y(2)*t);
case '9'
z=sin(x(3)*t)+sin(y(3)*t);
case '*'
z=sin(x(4)*t)+sin(y(1)*t);
case '0'
z=sin(x(4)*t)+sin(y(2)*t);
case '#'
z=sin(x(4)*t)+sin(y(3)*t);
otherwise
fprintf("invalid number");
end
sound(z)
subplot(n,2,i)
plot(t,z,'k')
xlabel("Samples (n)")
ylabel("Amplitude")
title(['Plot for ' num2str(num(i)) ' - 20071A04M1']);
pause(1);
end
```

Output : Enter number: 0123456789



Result : Thus, DTMF Signal is generated.

EXPERIMENT NO: 8

Implementation of Decimation and Interpolation processes, I/D sampling Rate Converters

Aim : To implement Decimation and Interpolation processes, I/D sampling Rate Converters using MATLAB software.

Software Used : MATLAB R2021b.

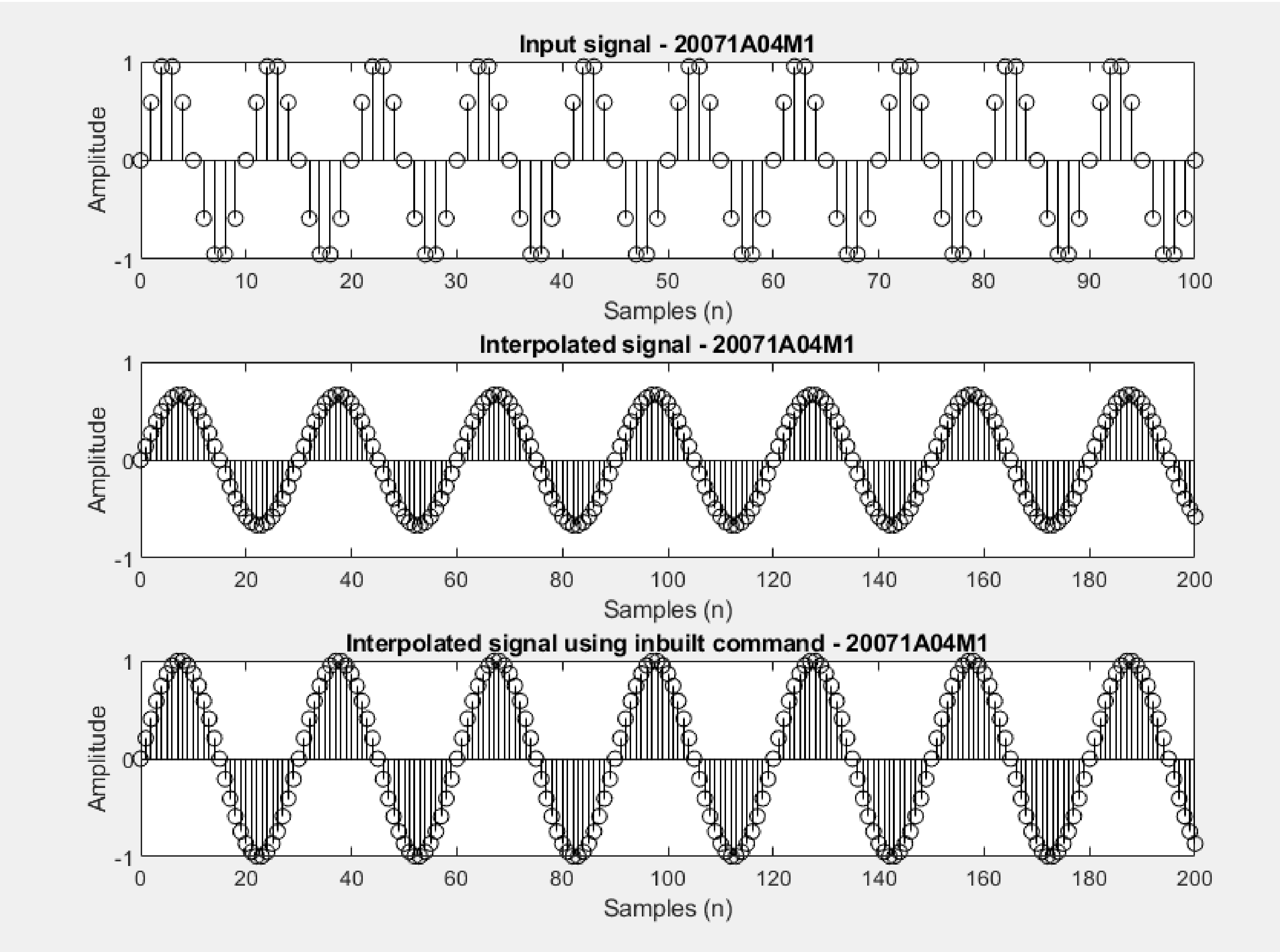
Program :

a) Interpolation :

```
clc
clear all
fs=100;
f=10;
L=input('Enter interpolation factor = ');
t=0:1/fs:1;
x=sin(2*pi*f*t);
N=length(x);
n=0:N-1;
m=0:(N*L)-1;
x1=zeros(1,L*N);
j=1:L:N*L;
x1(j)=x;
f1=fir1(34,0.48,'low');
z=2*filtfilt(f1,1,x1);
y=interp(x,L);
subplot(3,1,1);
stem(n,x,'k')
title('Input signal - 20071A04M1')
xlabel('Samples (n)')
ylabel('Amplitude')
subplot(3,1,2)
```

```
stem(m,z,'k')
axis ([0 200 -1 1])
title('Interpolated signal - 20071A04M1')
xlabel('Samples (n)')
ylabel('Amplitude')
subplot(3,1,3)
stem(m,y,'k')
axis ([0 200 -1 1])
title('Interpolated signal using inbuilt command - 20071A04M1')
xlabel('Samples (n)')
ylabel('Amplitude')
```

Output : Enter interpolation factor = 3



b)Decimation :

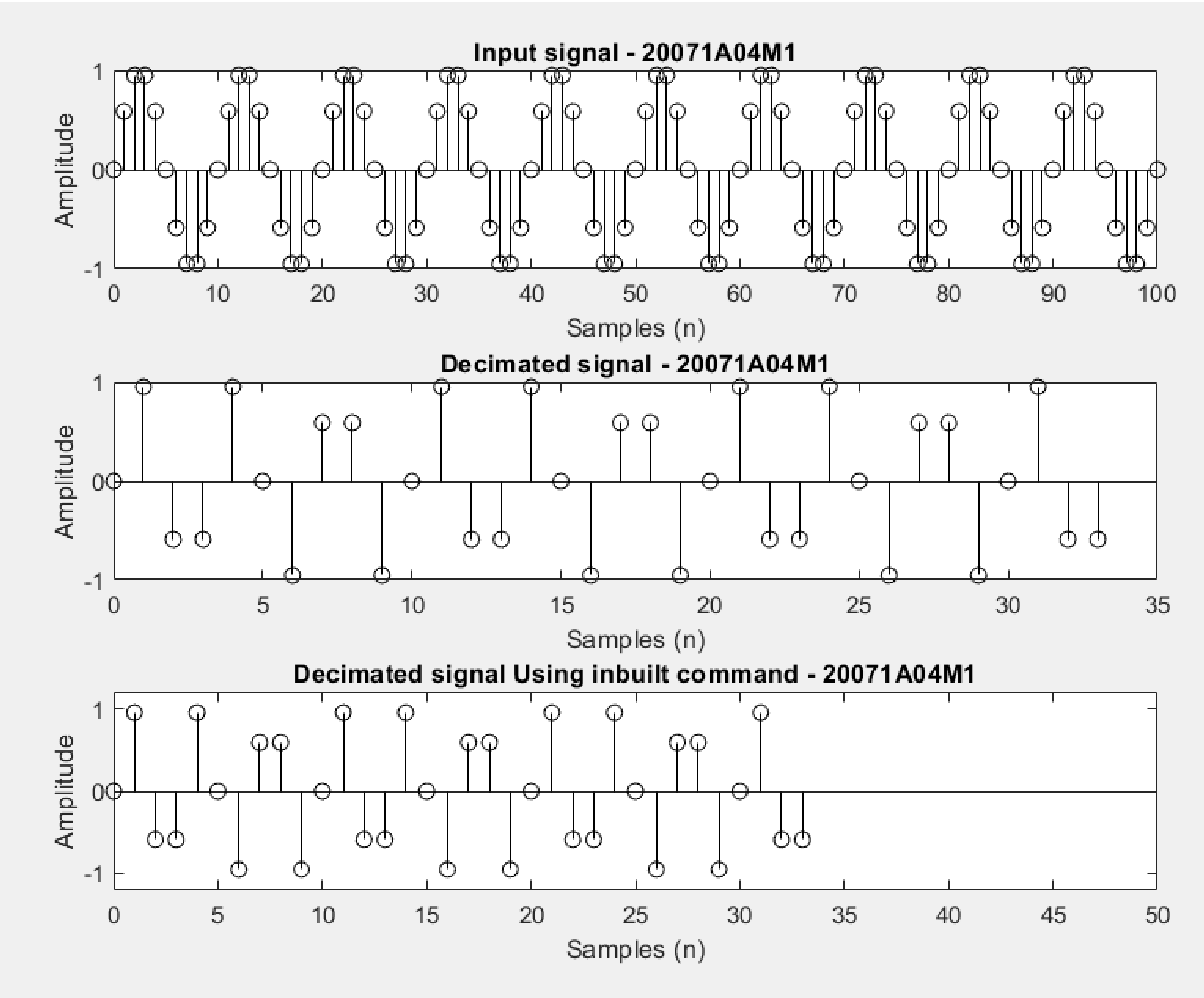
```
clc
clear all
fs=100;
fm=10;
D=input('enter decimation factor = ');
t=0:1/fs:1;
x=sin(2*pi*fm*t);
N=length(x);
n=0:N-1;
m=0:(N/D);
y=zeros(1,length(m));
j=1:D:N;
y=x(j);
v=decimate(x,D,'fir');
subplot(3,1,1)
stem(n,x,'k')
title('Input signal - 20071A04M1');
xlabel('Samples (n)')
ylabel('Amplitude')
subplot(3,1,2)
stem(m,y,'k')
title('Decimated signal - 20071A04M1')

xlabel('Samples (n)')
ylabel('Amplitude')
subplot(3,1,3)
stem(m,v,'k')
axis([0 50 -1.2 1.2])
title('Decimated signal Using inbuilt command - 20071A04M1')
xlabel('Samples (n)')
```

```
ylabel('Amplitude')
```

Output :

enter decimation factor = 3



c) I/D sampling Rate Converters :

```
clc
clear all
L=input('enter the upsampling factor = ');
D=input('enter the downsampling factor = ');
N=input('enter the length of the input signal = ');
f1=input('enter the frequency of first sinusoidal = ');
n=0:N-1;
x=sin(2*pi*f1*n);
y=resample(x,L,D);
subplot(2,1,1)
stem(n,x(1:N),'k')
title('Input sequence - 20071A04M1')
xlabel('Samples (n)')
ylabel('Ampitude')
subplot(2,1,2)
m=0:N*L/D-1;
stem(m,y(1:N*L/D),'k')
title('Output sequence - 20071A04M1')
xlabel('Samples (n)')
ylabel('Amplitude')
```

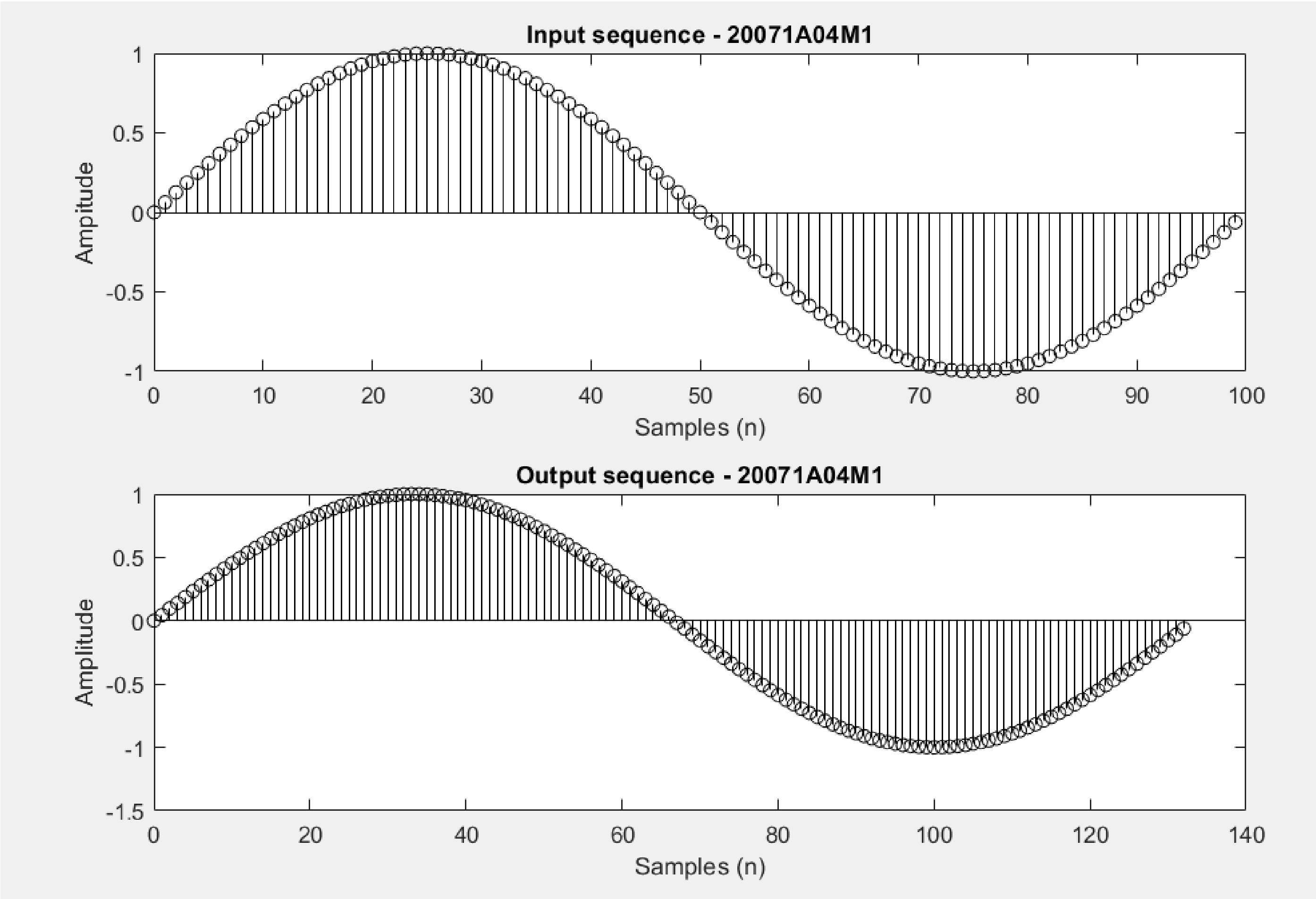
Output :

enter the upsampling factor = 4

enter the downsampling factor = 3

enter the length of the input signal = 100

enter the frequency of first sinusoidal = 0.01



Result : Thus, Decimation and Interpolation processes, I/D sampling Rate Converters are implemented.

CODE COMPOSER STUDIO STEPS

• Start CCS setup by double clicking on the CCS desktop icon.

New project creation:

• To create a new project, choose new CSS project from the project menu.

• Then Select the target as **C671X Floating point DSP** and click on **TMS320C6713**

Enter the project name.

Then Click on Finish.

New Target configuration:

• In View Option, select Target configuration.

• Click on the project folder and select New Target configuration.

• Then change the filename as **6713.ccxml**.

• Click on Finish.

• In basic window, select Connection as **Texas Instruments Simulator**.

select Device as **C67xx CPU Cycle Accurate Simulator, little Endian**.

-Click on Save.

• Then in Advanced window, select **TMS320C67XX.**

• Then in CPU Properties window click on browse button and open **DSK6713.gel**

file.

-Click on Save.

Launch Target configuration:

• Click on the User defined folder and select **C713.ccxml**.

Click on Launch target configuration.

Then click on the stop button.

• Then a CSS Editor window will be opened.

Click on your project and then click on main.c to type the programs.

Execution:

- To compile a program, right click on the project name.

Select build configuration and then click on build all.

- To run a program, click on the debug option in run menu.

Click on the run option again to display the output in the console window.

Experiment -1

LINEAR CONVOLUTION

Aim : To Perform Linear Convolution of two sequences using C-Code.

Software Used : Code Composer Studio

Program :

```
#include<stdio.h>
void main()
{
int a[9]={1,2,3,4,5};
int b[9]={1,1,1,1,1};
int m=5,n=5;
int c[10],i,j;
for(i=0;i<m+n-1;i++)
{
 c[i]=0;
 for(j=0;j<=i;j++)
 c[i]=c[i]+(a[j]*b[i-j]);
}
for(i=0;i<m+n-1;i++)
printf ("%d", c[i]);
}
```

Output :

1 3 6 10 15 14 12 9 5

Result : Thus, Linear Convolution of two sequences is performed using C-Code.

Experiment - 2

CIRCULAR CONVOLUTION

Aim : To Perform Circular Convolution of two sequences using C-Code

Software Used : Code Composer Studio

Program :

```
#include<stdio.h>
#define MAX 10
void main( )
{
int a[MAX],b[MAX],i,j,M,N,temp,k,c[MAX],x,z;
int max(int ,int );
printf("ENTER THE RANGE OF THE ARRAY A:(MAX=10)\n");
scanf("%d",&M);
printf("ENTER THE RANGE OF THE ARRAY B:(MAX=10)\n");
scanf("%d",&N);
printf("ENTER THE ELEMENTS OF THE ARRAY A:\n");
for(i=0;i<M;i++)
scanf("%d",&a[i]);
printf("ENTER THE ELEMENTS OF THE ARRAY B:\n");
for(i=0;i<N;i++)
scanf("%d",&b[i]);
k=max(M,N);
if(k==M)
{
for(i=N;i<M;i++)
b[i]=0;
for(i=0;i<M;i++)
```

```
c[i]=b[M-1-i];
}
else
{
for(i=M;i<N;i++)
a[i]=0;
for(i=0;i<N;i++)
c[i]=a[N-i-1];
}
printf("THE VALUES OF THE ARRAY (A CIRCULAR CONV B):\n");
for(i=0;i<k;i++)
{
temp=c[0];
c[0]=c[k-1];
for(z=k-1;z>=2;z--)
{
c[z]=c[z-1];
}
c[1]=temp;
x=0;
for(j=0;j<k;j++)
{
if(k==M)
x=x+a[j]*c[j];
else
x=x+b[j]*c[j];
}
printf("%d ",x);
}
```

```
}
int max(int l,int p)
{
if(l>p)
return(l);
else
return(p);
}
```

Output :

ENTER THE RANGE OF THE ARRAY A:(MAX=10)

4

ENTER THE RANGE OF THE ARRAY B:(MAX=10)

3

ENTER THE ELEMENTS OF THE ARRAY A:

1

2

3

4

ENTER THE ELEMENTS OF THE ARRAY B:

1

2

3

THE VALUES OF THE ARRAY (A CIRCULAR CONV B):

18 16 10 16

Result : Thus, Circular Convolution of two sequences is performed using C-Code.

Experiment -3

CORRELATION

Aim : To Perform correlation of two sequences using C-Code

Software Used : Code Composer Studio

Program :

```
#include<stdio.h>
#define MAX 10
void main()
{
int a[MAX],b[MAX],i,j,M,N,temp;
printf("ENTER THE RANGE OF THE ARRAY A:(MAX=10)\n");
scanf("%d",&M);
printf("ENTER THE RANGE OF THE ARRAY B:(MAX=10)\n");
scanf("%d",&N);
printf("ENTER THE ELEMENTS OF THE ARRAY A:\n");
for(i=0;i<M;i++)
scanf("%d",&a[i]);
printf("ENTER THE ELEMENTS OF THE ARRAY B:\n");
for(i=N-1;i>=0;i--)
scanf("%d",&b[i]);
printf("THE VALUES OF THE ARRAY (A COR B):\n");
for(i=0;i<(M+N-1);i++)
{
temp=0;
for(j=0;j<=i;j++)
{
if((j<M)&&((i-j)<N))
temp+=a[j]*b[i-j];
}
```

```
printf("%d ",temp);
}
}
```

Output :

ENTER THE RANGE OF THE ARRAY A:(MAX=10)

4

ENTER THE RANGE OF THE ARRAY B:(MAX=10)

3

ENTER THE ELEMENTS OF THE ARRAY A:

1 2 5 6

ENTER THE ELEMENTS OF THE ARRAY B:

8 9 1

THE VALUES OF THE ARRAY (A COR B):

1 11 31 67 94 48

Result : Thus, Correlation of two sequences is performed using C-Code.

Experiment - 5

GENERATION OF SINE WAVE

Aim : To Generate sine wave using C-Code

Software Used : Code Composer Studio

Program :

```
#include<stdio.h>
#include<math.h>
float a[100];
main()
{
int i;
for(i=0;i<11;i++)
{
a[i]= sin(2*3.14*5*i/100);
printf("%f",a[i]);
}
}
```

Output :

0.0000000

0.3088660

0.5875280

0.8087360

0.9508591

0.0000000

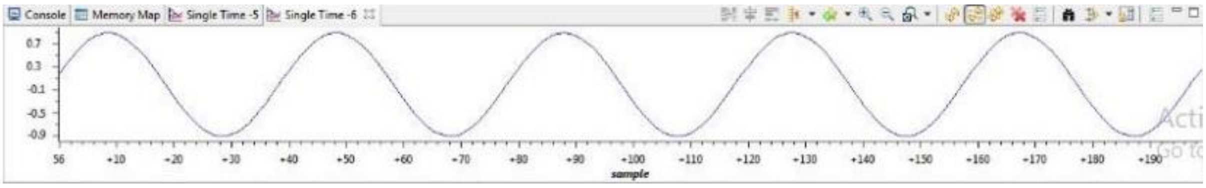
0.9513510

0.8096720

0.5888160

0.3103800

0.001593



Result : Thus, Sine Wave is generated using C-Code.

Experiment – 5

Discrete Fourier Transform

Aim : To Perform DFT using C - Code

Software Used : PC and Code Composer Studio

Program :

```
#include<stdio.h>
#include<math.h>
int N,k,n,i;
float pi=3.1416,sumre=0, sumim=0,out_real[8]={0.0}, out_imag[8]={0.0};
int x[32];
void main(void)
{
printf(" enter the length of the sequence\n");
 scanf("%d",&N);
 printf(" enter the sequence\n");
 for(i=0;i<N;i++)
 scanf("%d",&x[i]);
for(k=0;k<N;k++)
{
sumre=0;
sumim=0;
for(n=0;n<N;n++)
{
sumre=sumre+x[n]* cos(2*pi*k*n/N);
sumim=sumim-x[n]* sin(2*pi*k*n/N);
}
out_real[k]=sumre;
out_imag[k]=sumim;
printf("X([%d])=\t%f\t+\t%fi\n",k,out_real[k],out_imag[k]);
```

```
    }
    }
```

Output :

enter the length of the sequence

4

enter the sequence

1 4 6 7

X([0])= 18.000000 + 0.000000i

X([1])= -4.999938 + 3.000043i

X([2])= -4.000000 + 0.000096i

X([3])= -5.000184 + -2.999868i

Result : Thus, DFT is performed using C - Code.